変える力を、ともに生み出す。
NTT DATAグループ

Live Migration Support

11th Nov. 2010

NTTDATA Corporation Japan
Kei Masumoto and Masanori Itoh

INDEX

0. Live migration

- Providing the scheme to migrate running VM instances from a physical machine to others with:

  1. Almost no visible downtime
  2. No transaction loss

[conclusion]

1. Source is not necessary.
2. Destination can be specified.
3. Let the scheduler decide destination (but in this time, just follow admin's instruction)
4. To enable live migration be possible, check CPUID in /proc/cpuinfo. If its superset, we think nakajima-san.
5. To prevent concurrent request, some algorithm is necessary to scheduler (is it necessary to implement this for bexer release?) lock?

➢ Why live migration?

1. Maintenance

ex. Upgrade/installing the patches to hypervisors/BIOS.

ex. One of HDD volumes RAID / one of bonded NICs is out of order.

ex. Regular period maintenance.

2. Distributing high-load

ex. when many VM instances are running on a specific physical machine.

3. Saving power

VM instances are too much scattered, move VM instances to a physical machine!

The following 3 programs are inevitable to achieve live migration.

1. Live migration
2. Get a VM instances list running on the physical machine.
3. Provide info which physical machine still has enough machine resource.

(※ The above functionalities can be achieved through EC2/OpenStack API)

Example:

```
# nova-migrate --live   compute-node1  instance-1111  compute-node2
                           source            instance-id         destination
```

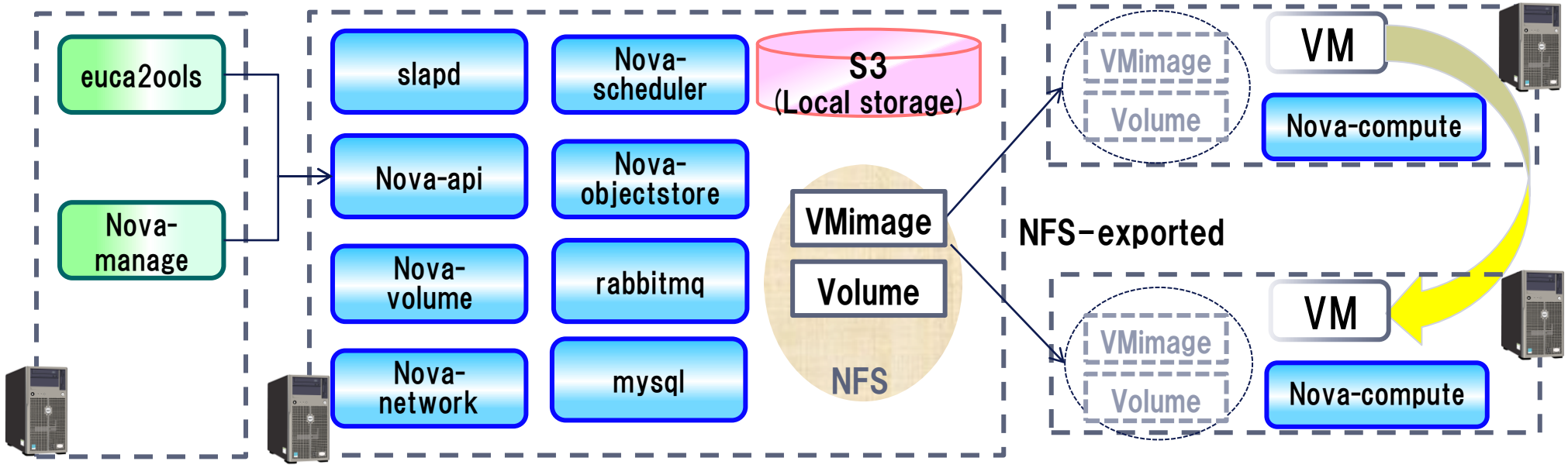
```
# nova-describe-instances
 INSTANCE i-45610761 0.0.0.0 0.0.0.0 pending 0 c1.medium 2010-04-01T04:44:10.774Z cluster1 eki-30D00D36 eri-8FC50F4E compute-node1
 INSTANCE i-45610761 emi-18240C94 0.0.0.0 0.0.0.0 pending 0 c1.medium 2010-04-01T04:44:10.774Z cluster1 eki-30D00D36 eri-8FC50F4E compute-node2
```

```
# nova-describe-physicalresource host1  ( no hostname given, getting info from all physical machine)
 TOTAL   host1                10      20480     1000    (total resource per physical machine)
 NODE    host1    project1     5      10240      100    ( usage per project)
 NODE    host1    project2     2       5012       30
 (nodename  - projectname - total cpu usage  - total memory usage - total hdd usage )
```

3. Requirements and Assumptions

	Explanation
OS	Ubuntu Lucid 10.04 (both physical machine and VM)
Hypervisor	KVM （ other hypervisors support may be discussed later)
Storages	The directory which instances are running and creating volumes must be a part of shared storage (using NFS)
Networks	Targeting nova.network.manager.VlanManager （ Other network managers support may be discussed later)
Network connectivity	Source /destination physical machines must belong to same availability zone (segment) .

Developing environment



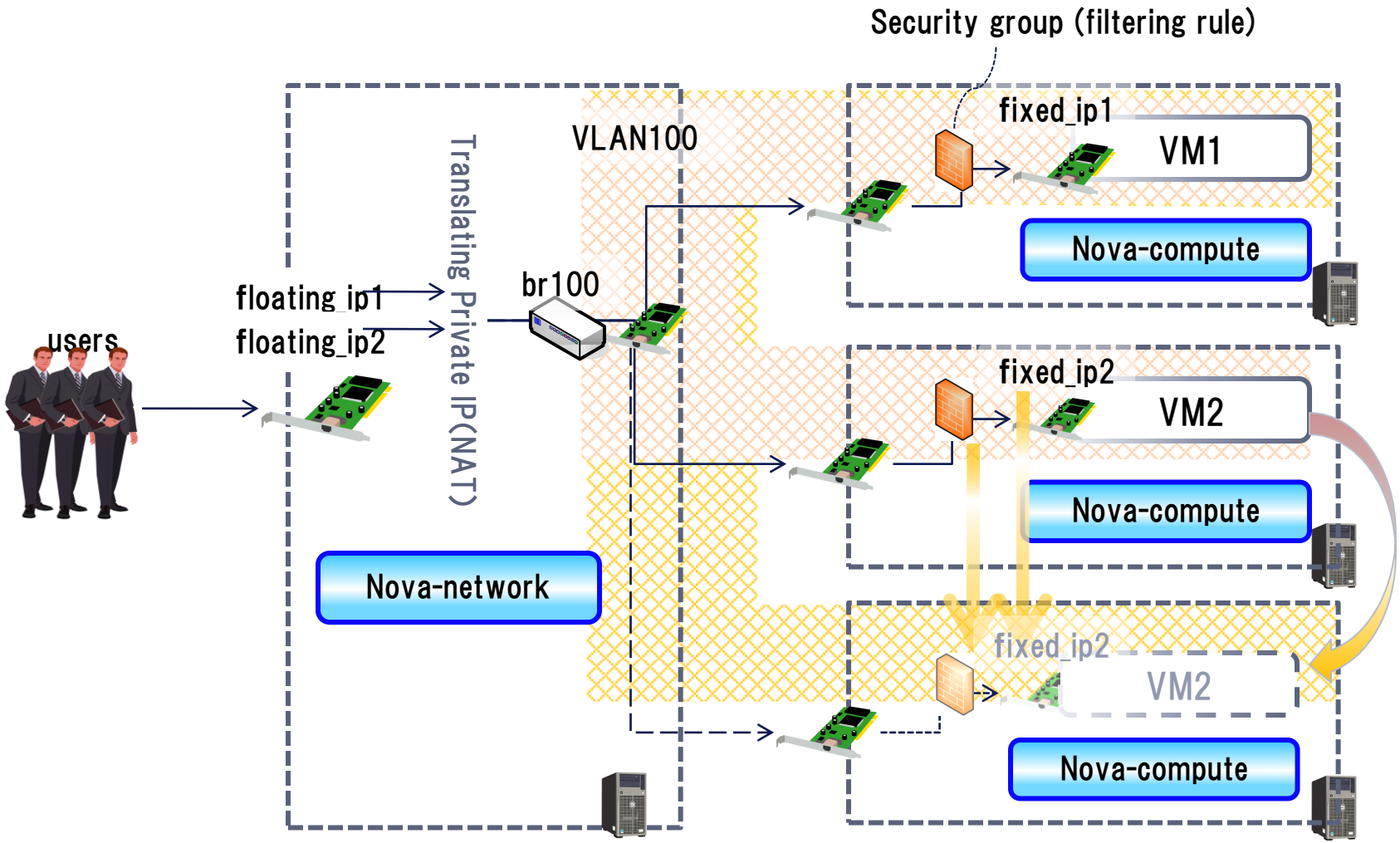
4. Other requirements

変える力を、ともに生み出す。
NTT DATAグループ

requirement	explanation
Authorization	Only administrators (not end user) must achieve live migration.
Deciding to destination machine	Administrator must decide which physical machine a VM instance moves to. (Future consideration may be system automatically decide destination, if it is necessary.)
Floating/Fixed_ip	VM instances must use same fix/floating ips.
EBS volume	VM instances must continue mounting volumes.
VLAN	VM instances must belong same VLAN.
Security group	VM instances must use same security group including filtering rules.

requirement	explanation
Test requirements	SSH connection is not terminated after live migration (High-load situation may be considered later)
Performance requirements	No requirements because it depends on storage/network performance.

VM instances can use same security group, floating/fixed ip and VLAN.



5. Design (live migration)

➢ How to execute

```
# nova-migrate  --live compute-node1   instance-111111  compute-node2
                        src                instance-id          dst
```

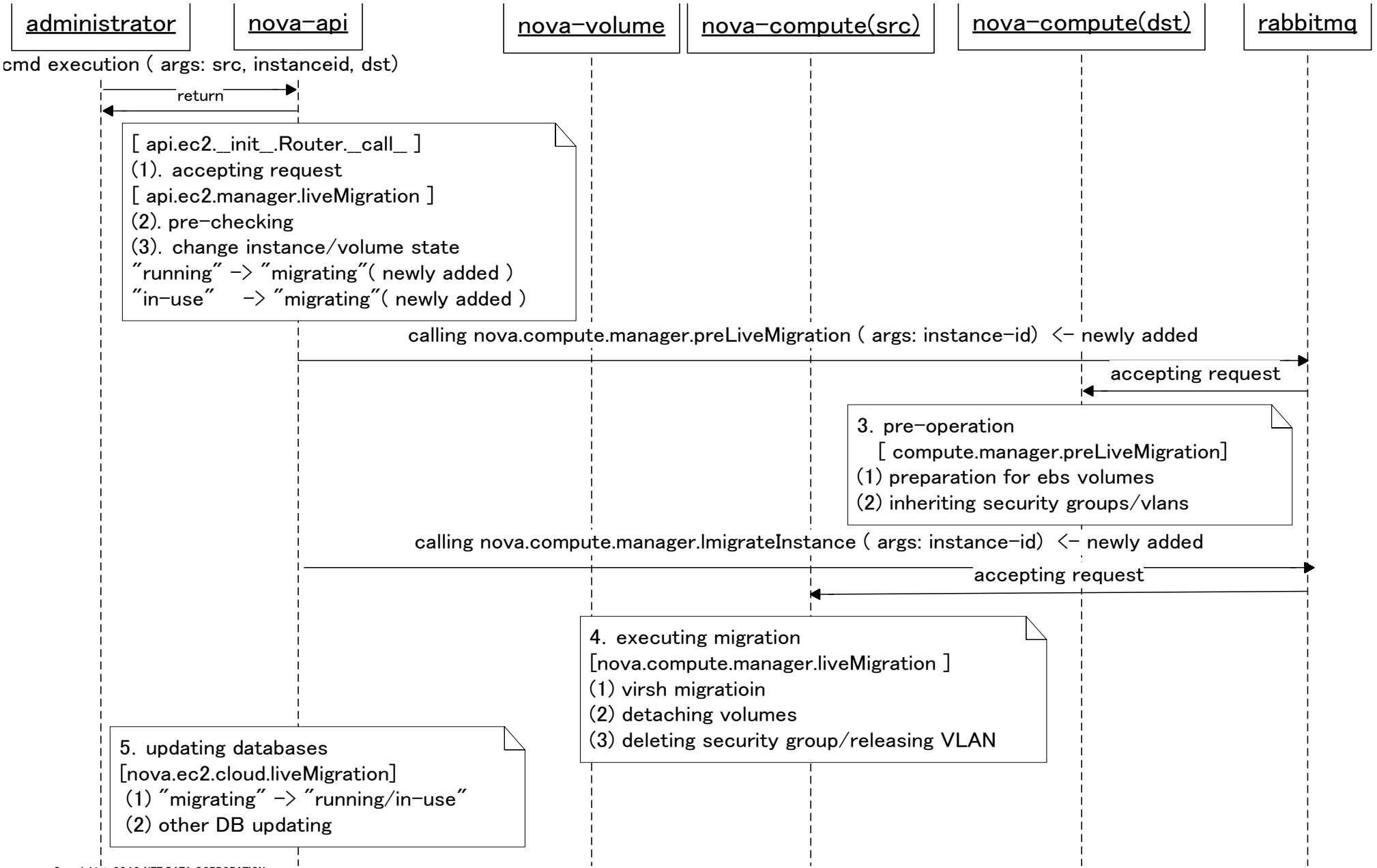
➢ Implementation

see next slide, make sure who can achieve each task.

➢ Discussion point

1. Which is better, implementing as a extension of nova-manage / a new command?
2. Which is better, implementing to a OpenStack API / EC2 API?
3. User has to be “Admin” and has to have a role “CLOUD ADMIN”.
4. Adding new state “migrating” is acceptable? (see next slide)

5. Design (live migration)



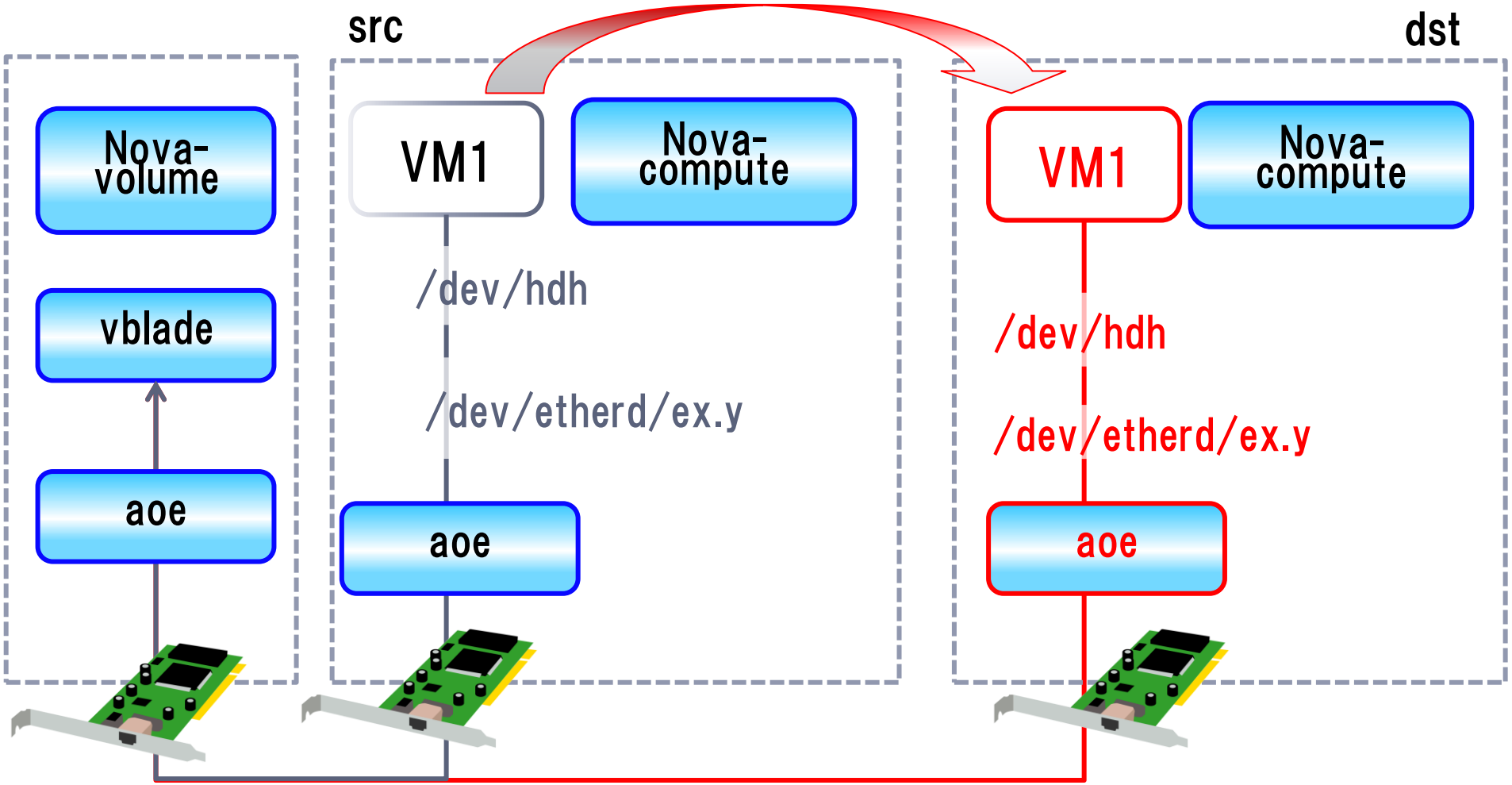
Supplement explanation

- **About 2. Pre-checking**
  1. Instance exists?/ instance status is available?
  2. Src/dst machine exist and up (up means nova.service.host_up)
  3. Dst machine has enough resource?

- **About 3. pre-operation for volumes**
  1. For keeping mounting same volume before/after live migration, we need preparation before starting live-migration (see next slide)

- **About 4. Executing migration**
  1. Planning to call “virsh migrate”.

The point which must be considered VM instance can keep attaching volumes.



5. Design (Getting list of instances and physical machines)

How to execute

```
# nova-describe-instances
 INSTANCE i-45610761 emi-18240C94 0.0.0.0 0.0.0.0 pending 0 c1.medium 2010-04-
01T04:44:10.774Z cluster1 eki-30D00D36 eri-8FC50F4E compute-node1
 INSTANCE i-45610761 emi-18240C94 0.0.0.0 0.0.0.0 pending 0 c1.medium 2010-04-
01T04:44:10.774Z cluster1 eki-30D00D36 eri-8FC50F4E compute-node2
```

Implementation

1. User executes above commands
2. Nova-api accepts the request, searching the “instance” table on DB, and getting information.

Discussion point

1. Which is better, implementing as an extension of nova-manage / a new command/ an extension of euca-describe-instance?
2. Which is better, implementing to a OpenStack API / EC2 API?
3. User has to be “Admin” and has to have a role “CLOUD ADMIN”

5. Design (getting physical machine resource)

How to execute

```
# nova-describe-physicalresource  host1  ( no hostname given, getting info from all physical machine)
 TOTAL   host1                      10        20480        1000    (total resource per physical machine)
 NODE    host1        project1       5        10240         100    ( usage per project)
 NODE    host1        project2       2         5012          30
(nodename  - projectname - total cpu usage  - total memory usage - total hdd usage )
```

Implementation

1. Users execute above command
2. nova-api accepts the request and gets any usage info per projects from the “instances” on DB.
3. nova-api sends request to nova-compute, and nova-compute
4. Nova-compute checks OS information, such as, /proc/cpuinfo for CPU, /proc/meminfo for memory, and /etc/mtab and FLAGS.instance_path for HDD.

Discussion point

1. Which is better, implementing as a extension of nova-manage / a new command?
2. Which is better, implementing to a OpenStack API / EC2 API?
3. User has to be “Admin” and has to be “CLOUD ADMIN”.

6. The policy for handling exceptions/errors.

- For unrecoverable errors (from Hypervisor point of view)
  1. Logging error messages.
  2. Doing the same way as terminating instances.

     (Delete any records about the instance from DB. If DB records for terminated instances remains for a while, please let me know.)

7. Future Considerations

The below considerations will not be included for Bexar release.
The discussions for future versions may be begun later.

- Other hypervisors support
- Other network managers support
- Using VPN considerations
- Block Migration（a live migration that shared storage is not necessary.）
- Other functionalities that newly added from Bexer release.

変える力を、ともに生み出す。

NTT DATAグループ